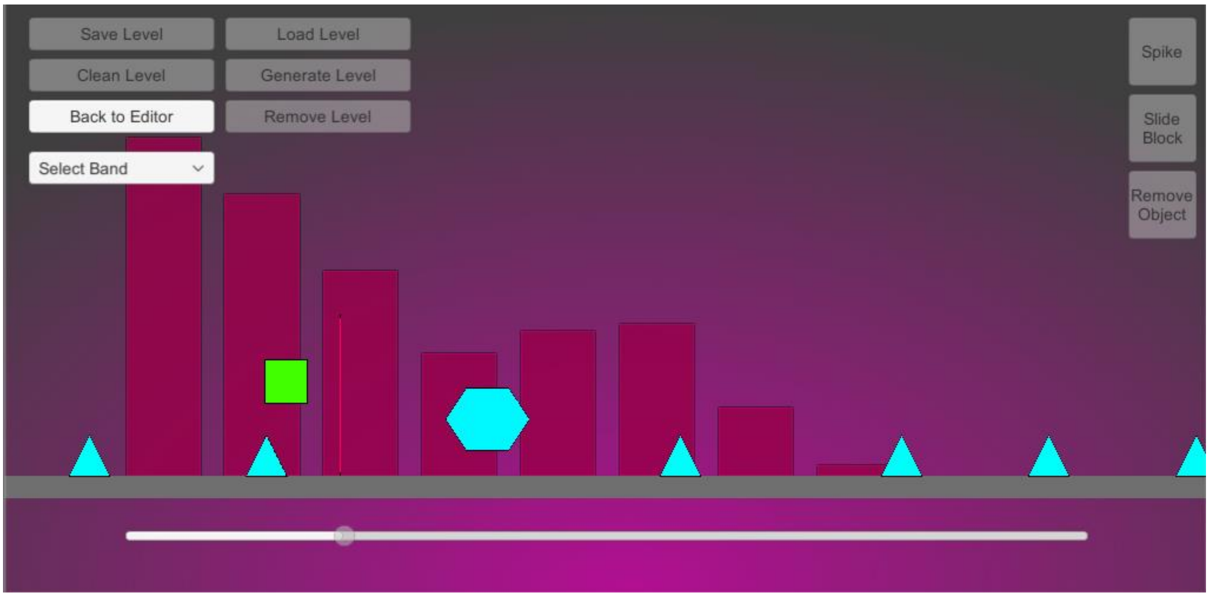
Read Me

Also available on: https://github.com/DanielQuarrell/CTP-MusicLevelGenerator

About

This project is a procedural level creation assistant tool that uses audio to generate levels for a 2D platformer. The aim of the project was to analyse music using audio processing techniques such as onset detection and use the output to influence procedural level generation.



Contents

0. Initial Set-up

1. Onset Visualiser Scene

2. Level Editor scene

3. Game scene

0. Initial Set-up

Open the project in Unity from the MusicLevelGenerator folder.

Ensure all audio files are stored in this directory:

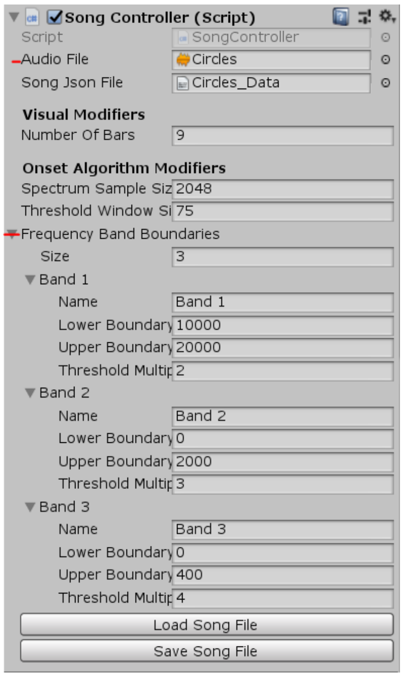
Assets/Resources/Audio

1. Onset Visualiser Scene

Before running the scene a song has to be set up for analysis.

The main interface is through the SongController gameobject/script – in the inspector window.

To start analysing a song place an audio file from the resources directory into the Audio File parameter and define the initial Frequency Band Boundaries.



Song Controller (Script)

Script: SongController

Audio File: Circles

Song Json File: Circles_Data

Visual Modifiers

Number Of Bars: 9

Onset Algorithm Modifiers

Spectrum Sample Siz: 2048

Threshold Window Si: 75

Frequency Band Boundaries

Size: 3

Band 1

Name: Band 1

Lower Boundary: 10000

Upper Boundary: 20000

Threshold Multip: 2

Band 2

Name: Band 2

Lower Boundary: 0

Upper Boundary: 2000

Threshold Multip: 3

Band 3

Name: Band 3

Lower Boundary: 0

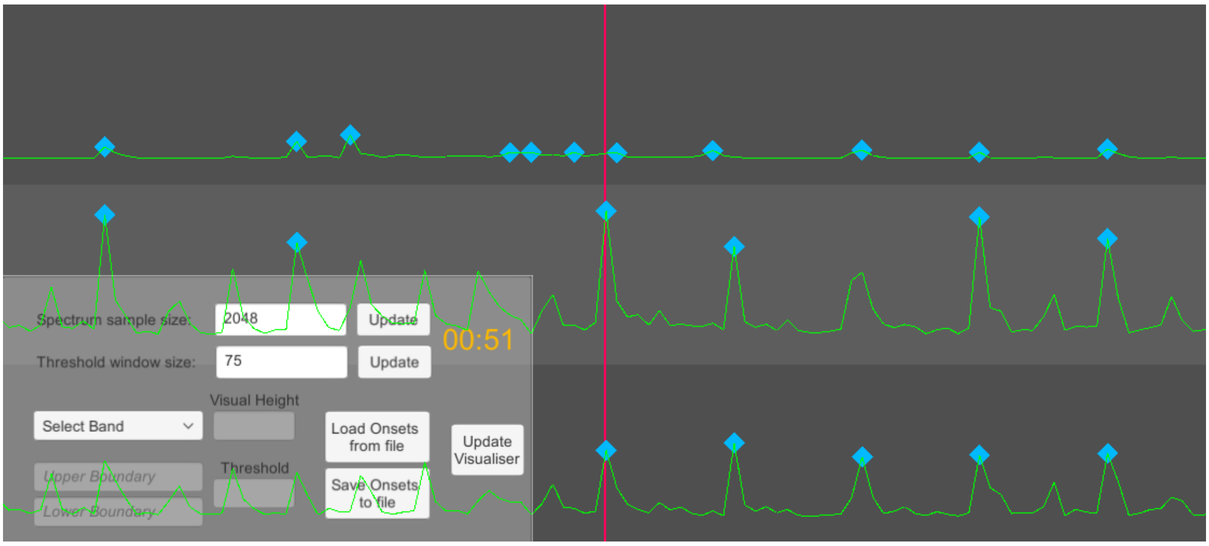
Upper Boundary: 400

Threshold Multip: 4

From here the scene can be played and the rest of the parameters can be modified in real time using the provided onscreen interface.

Note: Keeping the SongController gameobject selected in the Hierarchy window does affect performance during runtime, deselect it to prevent performance drops.

If any parameters are modified then the ‘Update Visualiser’ button will need to be pressed in order to re-process the song.

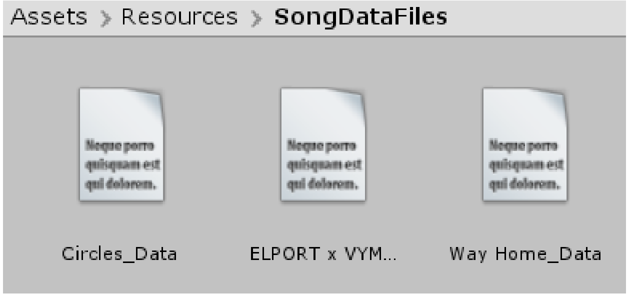


Spectrum sample size: 2048

Threshold window size: 75

To finish up, press the ‘Save the onsets to a file’ button which saves out a JSON file to this directory:

Assets/Resources/SongDataFiles/SongName_Data.JSON



This resulting JSON can be used to re-load the frequency bands back into the visualiser by placing it into the Song Json File parameter on the SongController and pressing either Load Song File (Editor) or Load Onsets from file(Runtime).

It is also required for using the Level Editor.

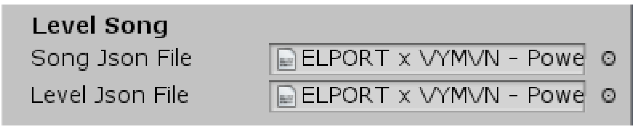
2. Level Editor scene

2.1 Setup

Before running the scene a song data file has to be set up to be used as the input for the level generation.

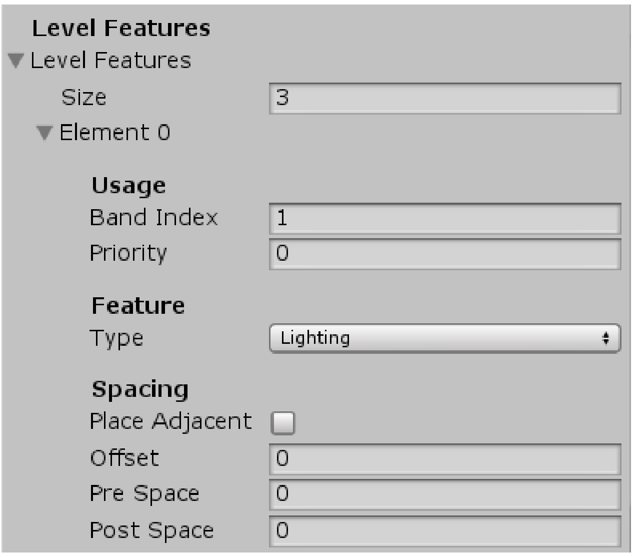
The main interface is through the LevelGenerator gameobject/script – in the inspector window.

First the song data file or 'SongName_Data.JSON' generated from the OnsetVisualiser scene needs to be placed in the Song Json File parameter.



Level Song
Song Json File: ELPORT x VYMVN - Powe
Level Json File: ELPORT x VYMVN - Powe

Scroll down to the Level Features array and assign level features to frequency bands, features with higher priorities (lowest number first) override features that have overlapping onsets.



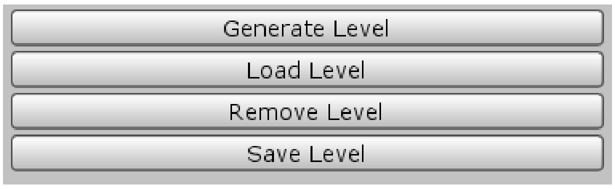
Level Features
▼ Level Features
Size: 3
▼ Element 0

Usage
Band Index: 1
Priority: 0

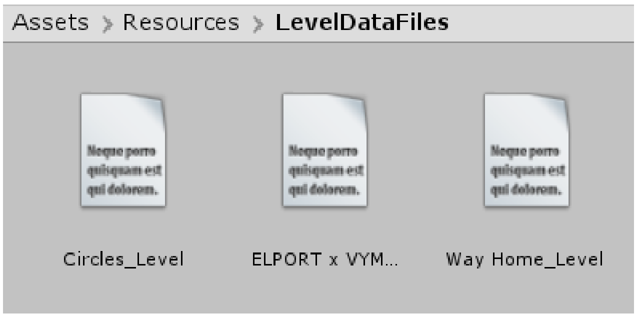
Feature
Type: Lighting

Spacing
Place Adjacent: □
Offset: 0
Pre Space: 0
Post Space: 0

Finally scroll down further to generate and save the level with the 'Generate Level' and 'Save Level' buttons in the inspector. This saves out a JSON file to this directory and automatically places the JSON in the LevelJsonFile ready for editing.



Assets/Resources/LevelDataFiles/SongName_Level.JSON



2.2 Editing

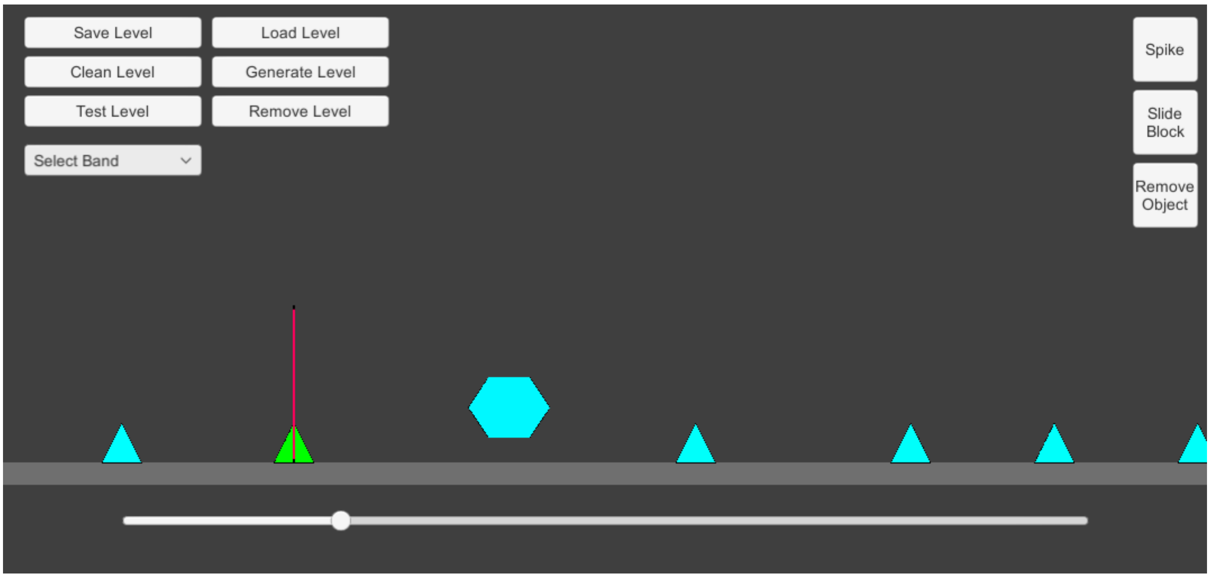
Play the scene to edit the current level.

Note: Keeping the LevelGenerator gameobject selected in the Hierarchy window does affect performance during runtime, deselect it to prevent performance drops.

The slider at the bottom of the screen can be used for quick navigation of the level and reflects level progress.

For more precise navigation use the 'A/D' or the 'LeftArrow/RightArrow' keys.

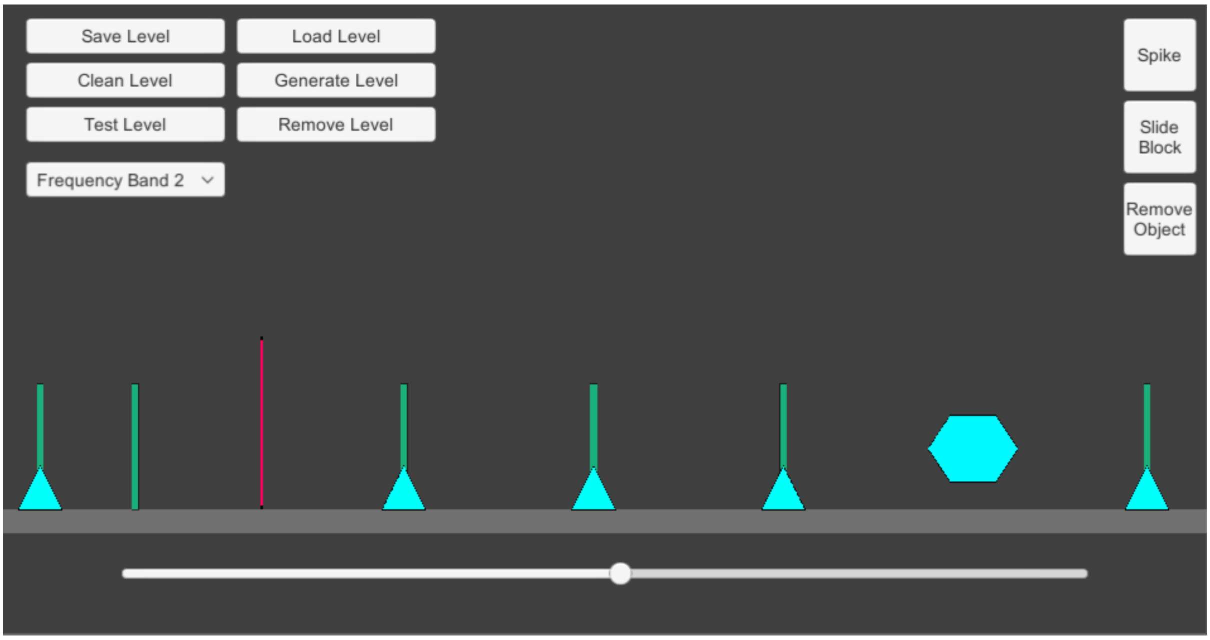
The red bar always represents the current song position in time. By moving it over a level object, it will highlight green and become selected. Pressing the 'remove object' button will remove the currently selected object.



To test the level, press the ‘Test Level’ button. This will start the level at the current position of the marker and respawns the player at that position if they hit an object.

Reference 3.2 Gameplay for instructions of how to play the game.

The ‘Select Band’ dropdown will create guideline markers representing the position of onsets from that frequency band. This can be used to place or replace objects where needed.



To finish up, press the ‘Save Level’ button which overwrites the JSON file with the modified level.#

3. Game scene

3.1 Set Up

This scene can be used to play the completed level.

The main interface is through the LevelGamplay gameobject/script – in the inspector window.

Drag and drop a Level JSON file from: Assets/Resources/LevelDataFiles/SongName_Level.JSON into the Level Json File parameter.

Run the scene.

3.2 Gameplay

- Press Space key to Jump
- Press the ‘S’ or ‘Down Arrow’ key to slide for a limited time
- Avoid obstacles for the duration of the song to win